**简介**

ThinkPHP是一个快速、简单的基于MVC和面向对象的轻量级PHP开发框架，遵循Apache2开源协议发布，从诞生以来一直秉承简洁实用的设计原则，在保持出色的性能和至简的代码的同时，尤其注重开发体验和易用性，并且拥有众多的原创功能和特性，为WEB应用开发提供了强有力的支持。

**目录结构**

ThinkPHP最新版本可以在官方网站（<http://thinkphp.cn/down/framework.html>）或者Github（<https://github.com/liu21st/thinkphp/downloads>）下载。  
把下载后的压缩文件解压到你的WEB目录（或者任何目录都可以），框架的目录结构为：

1. ├─ThinkPHP.php     框架入口文件
2. ├─Common 框架公共文件
3. ├─Conf 框架配置文件
4. ├─Extend 框架扩展目录
5. ├─Lang 核心语言包目录
6. ├─Lib 核心类库目录
7. │  ├─Behavior 核心行为类库
8. │  ├─Core 核心基类库
9. │  ├─Driver 内置驱动
10. │  │  ├─Cache 内置缓存驱动
11. │  │  ├─Db 内置数据库驱动
12. │  │  ├─TagLib 内置标签驱动
13. │  │  └─Template 内置模板引擎驱动
14. │  └─Template 内置模板引擎
15. └─Tpl 系统模板目录

复制代码

注意，框架的公共入口文件**ThinkPHP.php**是不能直接执行的，该文件只能在项目入口文件中调用才能正常运行（后面会讲到），这是很多新手很容易犯的一个错误。

**入口文件**

在开始之前，你需要一个Web服务器和PHP运行环境，如果你暂时还没有，我们推荐使用集成开发环境[WAMPServer](http://www.wampserver.com/en)（是一个集成了Apache、PHP和MySQL的开发套件，而且支持多个PHP版本、MySQL版本和Apache版本的切换）来使用ThinkPHP进行本地开发和测试。  
接下来我们先在WEB根目录下面创建一个app子目录（这个app就是我们的项目名），然后在该目录下面创建一个index.php文件，添加一行简单的代码：

1. <?php
2. require '/ThinkPHP框架所在目录/ThinkPHP.php';

复制代码

这行代码的作用就是加载ThinkPHP框架的入口文件ThinkPHP.php，这是所有基于ThinkPHP开发应用的第一步。  
然后，在浏览器中访问这个入口文件。

1. http://localhost/app/

复制代码

一般Web服务器的默认文件是index.php，所以我们可以不需要在URL地址中加上index.php。运行后我们会看到欢迎页面，  
[](http://www.thinkphp.cn/Uploads/info/hello.jpg)  
而且已经自动生成了项目目录，目录结构如下：

1. ├─index.php     项目入口文件
2. ├─Common 项目公共文件目录
3. ├─Conf 项目配置目录
4. ├─Lang 项目语言目录
5. ├─Lib 项目类库目录
6. │  ├─Action Action类库目录
7. │  ├─Behavior 行为类库目录
8. │  ├─Model 模型类库目录
9. │  └─Widget Widget类库目录
10. ├─Runtime 项目运行时目录
11. │  ├─Cache 模板缓存目录
12. │  ├─Data 数据缓存目录
13. │  ├─Logs 日志文件目录
14. │  └─Temp 临时缓存目录
15. └─Tpl 项目模板目录

复制代码

如果你希望项目的入口文件移动到app目录的外面，那么只需要修改入口文件index.php的内容为：

1. <?php
2. define('APP\_NAME','app');
3. define('APP\_PATH','./app/');
4. require '/ThinkPHP框架所在目录/ThinkPHP.php';

复制代码

APP\_NAME和APP\_PATH分部用于定义项目名和项目目录，项目名通常就是指项目的目录名称。  
移动并修改完项目的入口文件后，我们就可以通过

1. http://localhost/

复制代码

访问app项目了。当然你也可以在Web根目录下面创建多个子目录来部署多个项目。

**调试模式**

ThinkPHP的运行模式包括调试模式和部署模式，默认情况下是运行在部署模式下面。部署模式下面性能优先，并且尽可能少地抛出错误信息，调试模式则以除错方便优先，关闭任何缓存，而且尽可能多的抛出错误信息，所以对性能有一定的影响。部署模式采用了项目编译机制，第一次运行会对核心和项目相关文件进行编译缓存，由于编译后会影响开发过程中对配置文件、函数文件和数据库修改的生效（除非你修改后手动清空Runtime下面的缓存文件）。因此为了避免以上问题，我们强烈建议新手在使用ThinkPHP开发的过程中**使用调试模式**，这样可以更好的获取错误提示和避免一些不必要的问题和烦恼。  
开启调试模式很简单，我们只需要在入口文件的开头加上一行常量定义代码：

1. <?php
2. define('APP\_DEBUG',TRUE); // 开启调试模式
3. require '/ThinkPHP框架所在目录/ThinkPHP.php';

复制代码

开发完成后，我们实际进行项目部署的时候，删除这行常量定义代码即可，或者改成：

1. define('APP\_DEBUG',false); // 关闭调试模式

复制代码

**配置**

每个项目都有一个独立的配置文件（位于项目目录的Conf/config.php），配置文件的定义格式均采用**PHP返回数组**的方式，例如：

1. // 项目配置文件
2. return array(
3. '配置参数'     => '配置值',
4. // 更多配置参数
5. //...
6. );

复制代码

一旦有需要，我们就可以在项目配置文件中添加相关配置项目。通常我们提到的添加配置项目，就是指在项目配置文件中添加：

1. '配置参数' => '配置值',

复制代码

配置值可以支持包括字符串、数字、布尔值和数组在内的数据，通常我们建议配置参数均使用大写定义。如果有需要，我们还可以为项目定义其他的配置文件。

**控制器**

需要为每个模块定义一个控制器类，控制器类的命名规范是：  
**模块名+Action.class.php** （模块名采用驼峰法并且首字母大写）  
系统的默认模块是Index，对应的控制器就是项目目录下面的Lib/Action/IndexAction.class.php，类名和文件名一致。默认操作是index，也就是控制器的一个public方法。初次生成项目目录结构的时候，系统已经默认生成了一个默认控制器（就是之前看到的欢迎页面），我们把index方法改成下面的代码：

1. class IndexAction extends Action {
2. public function index(){
3. echo 'hello,world!';
4. }
5. }

复制代码

控制器必须继承Action类，一个模块可以包括多个操作方法。如果你的操作方法是protected或者private类型的话，是无法直接通过URL访问到该操作的。

**URL请求**

入口文件是项目的单一入口，对项目的所有请求都定向到项目的入口文件，系统会从URL参数中解析当前请求的模块和操作，我们之前访问的URL地址中没有任何参数，因此系统会访问默认模块（Index）的默认操作（index），因此下面的访问和之前是等效的：

1. http://localhost/app/index.php/Index/index

复制代码

这种URL模式就是系统默认的PATHINFO模式，不同的URL模式获取模块和操作的方法不同，ThinkPHP支持的URL模式有四种：普通模式、PATHINFO、REWRITE和兼容模式。  
  
**普通模式**：也就是传统的GET传参方式来指定当前访问的模块和操作，例如：

1. http://localhost/app/?m=module&a=action&var=value

复制代码

m参数表示模块，a操作表示操作（模块和操作的URL参数名称是可以配置的），后面的表示其他GET参数。  
  
**PATHINFO模式**：是系统的默认URL模式，提供了最好的SEO支持，系统内部已经做了环境的兼容处理，所以能够支持大多数的主机环境。对应上面的URL模式，PATHINFO模式下面的URL访问地址是：

1. http://localhost/app/index.php/module/action/var/value/

复制代码

PATHINFO地址的第一个参数表示模块，第二个参数表示操作。  
PATHINFO模式下面，URL是可定制的，例如，通过下面的配置：

1. 'URL\_PATHINFO\_DEPR'=>'-', // 更改PATHINFO参数分隔符

复制代码

我们还可以支持下面的URL访问：

1. http://localhost/app/index.php/module-action-var-value/

复制代码

**REWRITE模式**：是在PATHINFO模式的基础上添加了重写规则的支持，可以去掉URL地址里面的入口文件index.php，但是需要额外配置WEB服务器的重写规则。  
如果是Apache则需要在入口文件的同级添加.htaccess文件，内容如下：

1. <IfModule mod\_rewrite.c>
2. RewriteEngine on
3. RewriteCond %{REQUEST\_FILENAME} !-d
4. RewriteCond %{REQUEST\_FILENAME} !-f
5. RewriteRule ^(.\*)$ index.php/$1 [QSA,PT,L]
6. </IfModule>

复制代码

接下来，就可以用下面的URL地址访问了：

1. http://localhost/app/module/action/var/value/

复制代码

**兼容模式**：是用于不支持PATHINFO的特殊环境，URL地址是：

1. http://localhost/app/?s=/module/action/var/value/

复制代码

兼容模式配合Web服务器重写规则的定义，可以达到和REWRITE模式一样的URL效果。

**视图**

ThinkPHP内置了一个编译型模板引擎，也支持原生的PHP模板，并且还提供了包括Smarty在内的模板引擎驱动。和Smarty不同，ThinkPHP在渲染模板的时候如果不指定模板，则会采用系统默认的定位规则，其定义规范是 **Tpl/模块名/操作名.html**，所以，Index模块的index操作的默认模板文件位于项目目录下面的Tpl/Index/index.html。  
例如：

1. <html>
2. <head>
3. <title>hello {$name}</title>
4. </head>
5. <body>
6. hello, {$name}!
7. </body>
8. </html>

复制代码

要输出视图，必须在控制器方法中进行模板渲染输出操作，例如：

1. class IndexAction extends Action {
2. public function index(){
3. $this->name = 'thinkphp'; // 进行模板变量赋值
4. $this->display();
5. }
6. }

复制代码

display方法中我们没有指定任何模板，所以按照系统默认的规则输出了Index/index.html模板文件。  
接下来，我们在浏览器中输入

1. http://localhost/app/

复制代码

浏览器中会输出  
hello,thinkphp!

**读取数据**

在开始之前，我们首先在数据库thinkphp中创建一个think\_data数据表（以mysql数据库为例）：

1. CREATE TABLE IF NOT EXISTS `think\_data` (
2. `id` int(8) unsigned NOT NULL AUTO\_INCREMENT,
3. `data` varchar(255) NOT NULL,
4. PRIMARY KEY (`id`)
5. ) ENGINE=MyISAM  DEFAULT CHARSET=utf8 ;
6. INSERT INTO `think\_data` (`id`, `data`) VALUES
7. (1, 'thinkphp'),
8. (2, 'php'),
9. (3, 'framework');

复制代码

如果我们需要读取数据库中的数据，就需要在项目配置文件中添加数据库连接信息如下：

1. // 添加数据库配置信息
2. 'DB\_TYPE'   => 'mysql', // 数据库类型
3. 'DB\_HOST'   => 'localhost', // 服务器地址
4. 'DB\_NAME'   => 'thinkphp', // 数据库名
5. 'DB\_USER'   => 'root', // 用户名
6. 'DB\_PWD'    => '', // 密码
7. 'DB\_PORT'   => 3306, // 端口
8. 'DB\_PREFIX' => 'think\_', // 数据库表前缀

复制代码

或者采用如下配置

1. 'DB\_DSN' => 'mysql://root@localhost:3306/thinkphp'

复制代码

使用DB\_DSN方式定义可以简化配置参数，DSN参数格式为：  
**数据库类型://用户名:密码@数据库地址:数据库端口/数据库名**  
如果两种配置参数同时存在的话，DB\_DSN配置参数优先。  
  
接下来，我们修改下控制器方法，添加读取数据的代码：

1. class IndexAction extends Action {
2. public function index(){
3. $Data = M('Data'); // 实例化Data数据模型
4. $this->data = $Data->select();
5. $this->display();
6. }
7. }

复制代码

这里用到了M函数，是ThinkPHP内置的实例化模型的方法，而且用M方法实例化模型不需要创建对应的模型类，你可以理解为M方法是直接在操作底层的Model类，而Model类具备基本的CURD操作方法。  
M('Data') 实例化后，就可以对think\_data数据表（think\_ 是我们在项目配置文件中定义的数据表前缀）进行操作（包括CURD）了，M函数的用法还有很多，我们以后会深入了解。  
定义好控制器后，我们修改模板文件，添加数据输出标签如下：

1. <html>
2. <head>
3. <title>Select Data</title>
4. </head>
5. <body>
6. <volist name="data" id="vo">
7. {$vo.id}--{$vo.data}<br/>
8. </volist>
9. </body>
10. </html>

复制代码

volist标签是内置模板引擎用于输出数据集的标签。{$vo.id} 和 {$vo.data} 的用法和Smarty类似，就是用于输出数据的字段，这里就表示输出think\_data表的id和data字段的值。  
我们访问

1. http://localhost/app/

复制代码

会输出

1. 1--thinkphp
2. 2--php
3. 3--framework

复制代码

**如果发生错误，检查你是否开启了调试模式或者清空Runtime目录下面的缓存文件。**  
**如果你看到了上面的输出结果，那么恭喜你已经拿到了入门ThinkPHP的钥匙！**

**总结**

本篇我们学习了ThinkPHP的目录结构、URL模式，如何创建项目的入口文件和开启调试模式，以及控制器、模板和模型的基础认识，后面会继续了解对数据的CURD操作。